Journal of the Text Encoding Initiative

Issue 2 | February 2012
Selected Papers from the 2010 TEI Conference

Transforming Backward

HTML and HTML+RDFa to TEI

Grant Leyton Simpson and Dot Porter

Electronic version
URL: http://journals.openedition.org/jtei/407
DOI: 10.4000/jtei.407
ISSN: 2162-5603

Publisher
TEI Consortium

Electronic reference
Grant Leyton Simpson and Dot Porter, « Transforming Backward », *Journal of the Text Encoding Initiative* [Online], Issue 2 | February 2012, Online since 03 February 2012, connection on 19 April 2019. URL : http://journals.openedition.org/jtei/407 ; DOI : 10.4000/jtei.407

Transforming Backward

HTML and HTML+RDFa to TEI

Grant Leyton Simpson and Dot Porter

AUTHOR'S NOTE

The authors would like to thank the two anonymous reviewers who reviewed this paper; their comments have greatly improved it. Any errors herein are the fault of the authors and not the reviewers.

1. Introduction

1 There are many benefits to using XSL Transformations (Clark 1999; Kay 2007) in digital document creation and management. One such benefit is that, for transformations for which both the source and target documents are well-formed XML (including XML application profiles such as TEI, DocBook, MathML, XHTML, KML, etc.),[1] the transformation has the potential to be bidirectional. As long as one can determine the proper relationships of elements or groups of elements between the source and target dialects, one can craft rules to match them. The possibility for bidirectional conversion provides affordances in preserving and editing born-digital documents and in linguistic analysis.[2]

2 The standard workflow for preparing digital editions for display is unidirectional, involving writing XSLT to transform TEI into either or both of the following:

- HTML and CSS
- XSL-FO (for conversion into a print friendly format, such as PDF).

3 With either method we implicitly recognize that TEI, even coupled with CSS, is not, for the most part, designed for direct presentation. In one sense, TEI documents that are transcriptions of non-digital source documents are themselves end products in terms of the transcription process. However, in terms of the contemporary electronic edition, TEI

XML is just one of many steps along the path toward generating the final product.[3] That is, TEI must be transformed "forward" to another format.[4] After this, other support structures, such as application code, must be built around it.

4 Capitalizing on the bidirectional possibilities for markup transformation, the authors experimented with converting HTML documents, both with and without embedded RDFa, [5] to TEI. We call this "transforming backward," in that converting HTML to TEI constitutes the opposite of our community's typical XSLT workflow. While conversion from XSL-FO to TEI would also be transforming backward, we chose to focus on HTML source documents because we believe that conversion from HTML to TEI (whether automatically or manually) is an increasingly important use case.

5 Many born-digital documents are encoded in formats that are, by design, meant for display, such as HTML.[6] Hypothetical future editions of such documents would most likely need to include a richer description of the document than that provided by a display-oriented format alone. That is, HTML lacks the rich vocabulary that TEI provides and thus is not ideal for document description of the sort that editors require. Thus we foresee cases where born-HTML documents could be supplemented and described by TEI in much the same way as TEI currently supplements and describes manuscripts and printed books. This use case is essentially one approach to the process of transcribing born-digital documents originally created for the web.

6 Transformation of HTML to another vocabulary would also benefit those who use TEI as a means for recording linguistic data. The accessibility and massive quantity of Web pages make them a rich source of data on contemporary language; these could be transcoded by means of XSLT for the purpose of building corpora to be mined or otherwise interpreted. For this reason, we also experiment with converting from HTML to TEI through an intermediate program to tag linguistic aspects of the text.

2. Methods and Results

7 We prepared three proof-of-concept transformations: one representing a data-centric HTML+RDFa document, one representing a more narrative HTML document, and one representing an HTML document to be incorporated into a linguistic corpus. We wrote and processed the transformations for all three examples using version 12 of the oXygen XML Editor (SyncRO Soft 2010), with Saxon-EE as the processor (Kay 2011). We used XSLT 2.0 due to its built-in `tokenize()` function.

2.1. Data-centric Document

8 The data-centric document is adapted from an example in the World Wide Web Consortium's "RDFa Primer" (Adida and Birbeck 2008, sec. 3.1):

```
<html xmlns="http://www.w3.org/1999/xhtml"
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  version="XHTML+RDFa 1.0" xml:lang="en">
  <head>
    <title>Example 1</title>
  </head>
  <body>
    <div typeof="foaf:Person" xmlns:foaf="http://xmlns.com/foaf/0.1/">
      <p property="foaf:name">Alice Birpemswick</p>
      <p>Email: <a rel="foaf:mbox"
href="mailto:alice@example.com">alice@example.com</a></p>
    </div>
  </body>
</html>
```

9 RDFa allows document authors to refine the semantic capabilities of HTML. In HTML, `<p>` and `<div>` are meant to be relatively flexible containers for information or narrative content. As such, they do not themselves carry much intrinsic meaning. The RDFa embedded within them in the example makes use of the Friend of a Friend (FOAF) vocabulary (Brickley 2000) to indicate to human and machine readers what kind of data the elements contain. That is, the div, which represents a person, contains a p that contains the person's name and a link that contains the person's e-mail.

10 We crafted an XSL transformation that included the following template designed to match the HTML and RDFa in the example:

```
<xsl:template match="//html:div[@typeof='foaf:Person']">
  <listPerson>
    <xsl:for-each select="//html:div[@typeof='foaf:Person']">
      <person>
        <persName>
          <xsl:value-of select="html:p[@property='foaf:name']"/>
        </persName>
        <email>
          <xsl:value-of select="html:p/a[@rel='foaf:mbox']"/>
        </email>
      </person>
    </xsl:for-each>
  </listPerson>
</xsl:template>
```

11 This constructs a TEI `listPerson` if one or more HTML `divs` with `@typeof='foaf:Person'` are present. Then, for each of these divs, it constructs a `person` with a `persName` whose value is selected from the value of a `p` (with

`@property='foaf:name'`) within the `div` and an `email` whose value is selected from the value of another `p` (with `@rel='foaf:mbox'`).[7]

12 The transformation of this HTML document produces a TEI document whose body consists of the following:[8]

```
<body>
  <listPerson>
    <person>
      <persName>Alice Birpemswick</persName>
      <email>alice@example.com</email>
    </person>
  </listPerson>
</body>
```

2.2. Narrative Document

13 For a more narrative example of HTML—that is, HTML without much in the way of data and without RDFa—we chose a snippet from the Project Gutenberg (hereafter PG) HTML version of Lewis Carroll's Alice's Adventures in Wonderland (Carroll 2008):[9]

```
<p>'Talking of axes,' said the Duchess, 'chop off her head!'</p>
<p>Alice glanced rather anxiously at the cook, to see if she meant to take
the hint; but the cook was busily stirring the soup, and seemed not to
be listening, so she went on again: 'Twenty-four hours, I THINK; or is
it twelve? I&mdash;'</p>
<p>'Oh, don't bother ME,' said the Duchess; 'I never could abide figures!'
And with that she began nursing her child again, singing a sort of
lullaby to it as she did so, and giving it a violent shake at the end of
every line:</p>
<pre>
   'Speak roughly to your little boy,
    And beat him when he sneezes:
   He only does it to annoy,
    Because he knows it teases.'

         CHORUS.

 (In which the cook and the baby joined):&mdash;

       'Wow! wow! wow!'
</pre>
```

14 This snippet presents challenges because it contains both prose and verse. At first it looks easy to transform. After all, most of the novel is merely prose and prose is rather easily transformed from HTML to TEI because of the equivalence between HTML <p> and TEI <p>. The XSLT would merely need to contain the following:[10]

```
<xsl:template match="//html:p">
  <p>
    <xsl:value-of select="normalize-space(.)"/>
  </p>
</xsl:template>
```

15 The verse sections are a bit more difficult. Since HTML does not contain any elements that indicate verse content, they are wrapped in HTML <pre> tags, indicating to the browser only that the text should be presented with whitespace intact. It is up to the human reader to understand this is verse because of the way it is presented. Secondly, as redacted by the creator of the PG version, the <pre> element above contains more than just verse. A metatextual "chorus" comment marks off the chorus of the silly little song from its verses and there is an aside that is also not part of the song—"(In which the cook and the baby both joined):"—but that appears within the <pre> block. Thus, one could say that all verse in this particular version of the text appears in <pre> tags, but not everything that appears in <pre> is verse.[11] This makes it difficult to use an XSLT statement to match HTML <pre> to TEI <lg>. The contents of <pre> in the HTML PG Alice are encoded as <pre> only so that the browser will render it with non-breaking spaces intact so that the reader will understand it as not prose. Were this preformatted text represented only as such in TEI, we would argue that the creator of the TEI document had not gone far enough in marking up the verse content of the novel, i.e. had not made use of basic elements such as <l>. Thus, encoding the verse content of the <pre> elements for PG's Alice requires either human coding of TEI or a script that would tokenize the lines of poetry (using the newline marker as the delimiter) and wrap the lines in <l>.

16 In order to account for the majority of <pre> sections within this document, which contain verse, we crafted the following template:[12]

```
<xsl:template match="//html:pre">
  <xsl:variable name="lines" select="tokenize(.,'n')"/>
  <xsl:for-each select="$lines">
    <xsl:if test="normalize-space(.)!=''">
      <l><xsl:value-of select="normalize-space(.)"/></l>
    </xsl:if>
  </xsl:for-each>
</xsl:template>
```

17 This template matches the HTML <pre> element in the document. It constructs an array, called lines, that contains all the lines of the content within the <pre>. Then, for each of the non-empty lines, it constructs a TEI <l> containing the content of that line (with space normalized). This is not intended to handle either the aside or non-verse uses of <pre> in the document but rather to provide an acceptable first-pass solution at encoding the verse sections as TEI.

18 The transformation results in a TEI document whose body consists of the following:

```
<body>
  <p>'Talking of axes,' said the Duchess, 'chop off her head!'</p>
  <p>Alice glanced rather anxiously at the cook, to see if she meant to
take the hint; but the cook was busily stirring the soup, and seemed not
to be listening, so she went on again: 'Twenty-four hours, I THINK; or is
it twelve? I—'</p>
  <p>'Oh, don't bother ME,' said the Duchess; 'I never could abide
figures!' And with that she began nursing her child again, singing a sort
of lullaby to it as she did so, and giving it a violent shake at the end
of every line:</p>
  <lg>
    <l>'Speak roughly to your little boy,</l>
    <l>And beat him when he sneezes:</l>
    <l>He only does it to annoy,</l>
    <l>Because he knows it teases.'</l>
    <l>CHORUS.</l>
    <l>(In which the cook and the baby joined):—</l>
    <l>'Wow! wow! wow!'</l>
  </lg>
</body>
```

2.3. Source Document for a Linguistic Corpus

19 The third document, the one intended to become part of a linguistic corpus, is a snippet from the PG version of Wilkie Collins's *Woman in White:*[13]

```
<P>
There, in the middle of the broad bright high-road&mdash;
there, as if it had that moment sprung out of the earth or
dropped from the heaven&mdash;stood the figure of a solitary
Woman, dressed from head to foot in white garments, her
face bent in grave inquiry on mine, her hand pointing to
the dark cloud over London, as I faced her.
</P>
```

20 TEI is particularly useful as a tool for linguistic analysis when the linguistic segment category elements[14], which allow researchers to build corpora to suit their needs, are used. We chose to focus here on part-of-speech (POS) tagging, a form of syntactic analysis, in order to provide a proof of concept.[15]

21 We produced TEI from the Collins snippet by means of a three-step process:

1. Running it through a POS tagger that was instructed to produce XML output
2. Replacing the character entity references produced by the tagger in step one with their proper characters
3. Transforming the results of step two to TEI by means of XSLT

22 To produce a tagged document, we invoked the Stanford Log-Linear Part-of-Speech Tagger, version 3.0.3 (Toutanova et al. 2011). We instructed it to accept XML input and produce XML output. The Collins fragment we chose was well-formed XML, so we did not have to run HTML Tidy (see note 2) on it before handing it over to the tagger. The model used was left3words-wsj-0-18, based on the *Wall Street Journal*; this model has a 96.97% accuracy rate ("Stanford POS tagger FAQ", n.d.).

23 Unfortunately, this output consisted mostly of an XML document that had been escaped by means of the entities `<`, `>`, and `"`.[16] Before applying our transformation, we replaced them with their proper forms. We also replaced uppercase <P> with lowercase <p>, but this was not strictly necessary, as we could have matched for <P> in the XSLT. The document, before XSLT processing, consisted of:

```
<body>
  <p>
    <sentence id="0">
      <word wid="0" pos="RB">There</word>
      <word wid="1" pos=",">,</word>
      <word wid="2" pos="IN">in</word>
      <word wid="3" pos="DT">the</word>
      <word wid="4" pos="NN">middle</word>
      <word wid="5" pos="IN">of</word>
      <word wid="6" pos="DT">the</word>
      <word wid="7" pos="JJ">broad</word>
      <word wid="8" pos="JJ">bright</word>
      <word wid="9" pos="JJ">high-road</word>
      <word wid="10" pos=":">--</word>
      <word wid="11" pos="RB">there</word>
      <word wid="12" pos=",">,</word>
      <word wid="13" pos="IN">as</word>
      <word wid="14" pos="IN">if</word>
      <word wid="15" pos="PRP">it</word>
      <word wid="16" pos="VBD">had</word>
      <word wid="17" pos="IN">that</word>
      <word wid="18" pos="NN">moment</word>
      <word wid="19" pos="VBN">sprung</word>
      <word wid="20" pos="IN">out</word>
      <word wid="21" pos="IN">of</word>
      <word wid="22" pos="DT">the</word>
      <word wid="23" pos="NN">earth</word>
      <word wid="24" pos="CC">or</word>
      <word wid="25" pos="VBD">dropped</word>
      <word wid="26" pos="IN">from</word>
      <word wid="27" pos="DT">the</word>
      <word wid="28" pos="NN">heaven</word>
      <word wid="29" pos=":">--</word>
      <word wid="30" pos="VBD">stood</word>
      <word wid="31" pos="DT">the</word>
      <word wid="32" pos="NN">figure</word>
      <word wid="33" pos="IN">of</word>
      <word wid="34" pos="DT">a</word>
      <word wid="35" pos="JJ">solitary</word>
      <word wid="36" pos="NNP">Woman</word>
      <word wid="37" pos=",">,</word>
      <word wid="38" pos="VBN">dressed</word>
      <word wid="39" pos="IN">from</word>
      <word wid="40" pos="NN">head</word>
      <word wid="41" pos="TO">to</word>
      <word wid="42" pos="NN">foot</word>
      <word wid="43" pos="IN">in</word>
      <word wid="44" pos="JJ">white</word>
      <word wid="45" pos="NNS">garments</word>
```

```
      <word wid="46" pos=",">,</word>
      <word wid="47" pos="PRP$">her</word>
      <word wid="48" pos="NN">face</word>
      <word wid="49" pos="NN">bent</word>
      <word wid="50" pos="IN">in</word>
      <word wid="51" pos="JJ">grave</word>
      <word wid="52" pos="NN">inquiry</word>
      <word wid="53" pos="IN">on</word>
      <word wid="54" pos="NN">mine</word>
      <word wid="55" pos=",">,</word>
      <word wid="56" pos="PRP$">her</word>
      <word wid="57" pos="NN">hand</word>
      <word wid="58" pos="VBG">pointing</word>
      <word wid="59" pos="TO">to</word>
      <word wid="60" pos="DT">the</word>
      <word wid="61" pos="JJ">dark</word>
      <word wid="62" pos="NN">cloud</word>
      <word wid="63" pos="IN">over</word>
      <word wid="64" pos="NNP">London</word>
      <word wid="65" pos=",">,</word>
      <word wid="66" pos="IN">as</word>
      <word wid="67" pos="PRP">I</word>
      <word wid="68" pos="VBD">faced</word>
      <word wid="69" pos="PRP">her</word>
      <word wid="70" pos=".">.</word>
    </sentence>
  </p>
</body>
```

24 The <body> and <p> elements in the document come from the original HTML document, whereas the new <sentence> and <word> elements were supplied by the tagger. The contents of <word>'s @pos correspond to the Penn Treebank II format (Santorini 1990).[17]

25 The XSLT we crafted to convert this to TEI consisted of three templates:

```
<xsl:template match="//temp:p">
  <p><xsl:apply-templates/></p>
</xsl:template>
<xsl:template match="temp:sentence">
  <s>
    <xsl:apply-templates/>
  </s>
</xsl:template>
<xsl:template match="temp:word">
  <w>
    <xsl:attribute name="type">
      <xsl:choose>
        <xsl:when test="matches(.,'w')"><xsl:value-of select="@pos"/></
xsl:when>
        <xsl:otherwise>PUN</xsl:otherwise>
      </xsl:choose>
    </xsl:attribute>
    <xsl:value-of select="."/>
  </w>
</xsl:template>
```

26 The first two templates merely match HTML <p> to TEI <p> and the Stanford tagger's <sentence> to TEI <s>, respectively. (We assigned the input document the namespace "temp," so both the HTML elements and the tagger elements appear under that namespace.) The third template matches the tagger's <word> to TEI <w> and @pos of each <word> to @type of <w>. In the cases where the tagger assigned a @pos consisting of a punctuation mark, we converted this to a @type that consists of "PUN."[18]

27 The final product of the language-oriented transformation is a TEI document whose body consists of the following:

```
<body>
  <p>
    <s>
      <w type="RB">There</w>
      <w type="PUN">,</w>
      <w type="IN">in</w>
      <w type="DT">the</w>
      <w type="NN">middle</w>
      <w type="IN">of</w>
      <w type="DT">the</w>
      <w type="JJ">broad</w>
      <w type="JJ">bright</w>
      <w type="JJ">high-road</w>
      <w type="PUN">--</w>
      <w type="RB">there</w>
      <w type="PUN">,</w>
      <w type="IN">as</w>
      <w type="IN">if</w>
      <w type="PRP">it</w>
      <w type="VBD">had</w>
      <w type="IN">that</w>
      <w type="NN">moment</w>
      <w type="VBN">sprung</w>
      <w type="IN">out</w>
      <w type="IN">of</w>
      <w type="DT">the</w>
      <w type="NN">earth</w>
      <w type="CC">or</w>
      <w type="VBD">dropped</w>
      <w type="IN">from</w>
      <w type="DT">the</w>
      <w type="NN">heaven</w>
      <w type="PUN">--</w>
      <w type="VBD">stood</w>
      <w type="DT">the</w>
      <w type="NN">figure</w>
      <w type="IN">of</w>
      <w type="DT">a</w>
      <w type="JJ">solitary</w>
      <w type="NNP">Woman</w>
      <w type="PUN">,</w>
      <w type="VBN">dressed</w>
      <w type="IN">from</w>
      <w type="NN">head</w>
      <w type="TO">to</w>
      <w type="NN">foot</w>
      <w type="IN">in</w>
      <w type="JJ">white</w>
      <w type="NNS">garments</w>
```

```
      <w type="PUN">,</w>
      <w type="PRP$">her</w>
      <w type="NN">face</w>
      <w type="NN">bent</w>
      <w type="IN">in</w>
      <w type="JJ">grave</w>
      <w type="NN">inquiry</w>
      <w type="IN">on</w>
      <w type="NN">mine</w>
      <w type="PUN">,</w>
      <w type="PRP$">her</w>
      <w type="NN">hand</w>
      <w type="VBG">pointing</w>
      <w type="TO">to</w>
      <w type="DT">the</w>
      <w type="JJ">dark</w>
      <w type="NN">cloud</w>
      <w type="IN">over</w>
      <w type="NNP">London</w>
      <w type="PUN">,</w>
      <w type="IN">as</w>
      <w type="PRP">I</w>
      <w type="VBD">faced</w>
      <w type="PRP">her</w>
      <w type="PUN">.</w>
    </s>
  </p>
</body>
```

3. Discussion

28 With the example transformations presented herein, we have demonstrated that transforming backward is possible. Under the right circumstances, this technique could prove fruitful for a project that needs to make use of existing HTML content. We anticipate that projects working with a large number of source documents might find that the benefits of automated or semi-automated transformation outweigh the costs of writing what may be quite project-specific XSLT. In addition, a tool such as OxGarage, which uses TEI as a "pivot format" (Oxford 2010) for conversions between multiple formats, could benefit from adding rules for matching RDFa (which can be hosted in multiple languages) in order to produce more nuanced translations.

29 We would like to stress that a backward transformation will, at times, need to be guided by and supplemented with human intelligence. Furthermore, transforming from HTML to TEI necessitates accounting for correspondences between HTML and TEI elements and attributes. The *Alice* example demonstrates that HTML authors' element choices can complicate the process: the element `<pre>` in *Alice* maintains the lineation of the poetry it contains but is unable to mark the content as poetry. Preformatted text is just one way that content authors get around an impoverished markup scheme, so such hacks and

other tag abuse can confer meaning to readers. It would be difficult to account for a significant number of these decisions and impossible to account for all of them.

30 In addition to accounting for element and attribute correspondence at the level of the markup language, transforming HTML+RDFa to TEI also requires accounting for:

1. All of the RDFa attributes, such as `@typeof`, `@property`, `@rel`, etc.
2. Correspondences between the vocabularies used (FOAF, Dublin Core, etc.) and TEI elements and attributes
3. Differences in the way relationships can be expressed by means of different combinations of HTML elements, RDFa attributes, and vocabulary choices

31 In addition to more experimentation with a generalized HTML-to-TEI stylesheet, we envision two main areas of further research in this area: developing RDF-aware transformations and expanding work to include HTML Microdata.

32 Whereas software that understands RDF understands the relationships among elements in an ontology (for example, the relationship between `foaf:Person` and `foaf:name`), XSLT does not and is not meant to. For example, the `@select` attribute of `<xsl:value-of select="p/a[@rel='foaf:mbox']"/>` does not instruct the XSL parser to find all the FOAF mbox items in the document. Rather, it instructs the parser to find each a element that is a child of a `p` element, and that has an `@rel` attribute, the value of which is the string "`foaf:mbox`". One could simplify the XPath statement to match merely any element that has `@rel='foaf:mbox'`, but this would not handle any differences in output that the stylesheet designer meant to produce based on the identity of the HTML element.[19] Furthermore, searching for an RDF predicate by means of its appearance as the value of a particular HTML+RDFa attribute, such as `@rel`, would handle cases where that predicate appeared in the same attribute for multiple HTML elements, but it would not handle instances of that predicate being expressed in a different attribute. We would like to see transformations that were in some sense aware of the RDF data model or at least able to make inferences about which attributes contain the same predicate. Such an "RDF-aware" transformation would then be able to employ common patterns based on the identity of the predicate.

33 On a less esoteric level, we would like to see development of conversions between HTML Microdata and TEI. Microdata (Hickson 2011) currently competes with RDFa as a method for making assertions about HTML content. At the time of this writing, Microdata is not yet a W3C recommendation. Despite this, the format received a major push when the three top search engines (Google, Bing, and Yahoo!) announced schema.org, a "one stop resource for webmasters looking to add markup to their pages" (Guha 2011) which promotes use of Microdata with the schema.org vocabulary to express information about page content (schema.org 2011).[20] It seems likely that Microdata usage, especially Microdata that uses the schema.org vocabulary, will increase. Converting HTML containing Microdata to TEI would be possible using the same techniques discussed above for converting HTML+RDFa to TEI. In fact, the same rules for matching TEI elements and attributes to a known vocabulary could be used because Microdata can express FOAF and other common vocabularies. If the schema.org vocabulary does gain widespread use, it would be worthwhile to devise a crosswalk between it and TEI. While it is debatable whether use of a single schema is a good thing for the web's development, it would drastically simplify the task of converting backward from HTML infused with Microdata to TEI.

BIBLIOGRAPHY

Adida, Ben and Mark Birbeck, eds. 2008. "RDFa Primer: Bridging the Human and Data Webs." World Wide Web Consortium. http://www.w3.org/TR/xhtml-rdfa-primer/.

Adida, Ben, Mark Birbeck, Shane McCarron and Steven Pemberton. 2008. "RDFa in XHTML: Syntax and Processing." World Wide Web Consortium. http://www.w3.org/TR/rdfa-syntax/.

Brickley, Dan. 2000. "Introducing FOAF." FOAF Project. http://www.foaf-project.org/original-intro.

Burnard, Lou and Syd Bauman. 2007. "P5: Guidelines for Electronic Text Encoding and Interchange." Text Encoding Initiative. http://www.tei-c.org/release/doc/tei-p5-doc/en/html/.

Carroll, Lewis. *Alice's Adventures in Wonderland* 2008. Produced by David Widger. Project Gutenberg. http://www.gutenberg.org/files/11/11-h/11-h.htm.

Clark, James. 1999. "XSL Transformations (XSLT) Version 1.0." World Wide Web Consortium. http://www.w3.org/TR/xslt.

Collins, Wilkie. 2008. *Woman in White*. Project Gutenberg. http://www.gutenberg.org/files/583/583-h/583-h.htm.

DuCharme, Bob. 2000. "HTML and XSLT." Xml.com. http://www.xml.com/pub/a/2000/08/30/xsltandhtml/index.html.

Guha, Ramanathan. 2011. "Introducing schema.org: Search Engines Come Together for a Richer Web." *The Official Google Blog*. http://googleblog.blogspot.com/2011/06/introducing-schemaorg-search-engines.html.

Hayes, Patrick. 2004. "RDF Semantics." World Wide Web Consortium. http://www.w3.org/TR/rdf-mt/.

Hickson, Ian. 2011. "HTML Microdata: W3C Working Draft 25 May 2011." World Wide Web Consortium. http://www.w3.org/TR/microdata/.

Kay, Michael, ed. 2007. "XSL Transformations (XSLT) Version 2.0." World Wide Web Consortium. http://www.w3.org/TR/xslt20/.

———. 2011. "Saxon: The XSLT and XQuery Processor." http://saxon.sourceforge.net/

Klyne, Graham and Jeremy J. Carroll. 2004. "Resource Description Framework (RDF): Concepts and Abstract Syntax." World Wide Web Consortium. http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.

Lassila, Ora and Ralph R. Swick. 1999. "Resource Description Framework (RDF) Model and Syntax Specification." World Wide Web Consortium. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.

Manola, Frank and Eric Miller. 2004. "RDF Primer." World Wide Web Consortium. http://www.w3.org/TR/rdf-syntax/.

Marchal, Benoît. 2003. "Tip: Convert from HTML to XML with HTML Tidy." IBM DeveloperWorks. http://www.ibm.com/developerworks/library/tiptidy.html.

Microformats. 2011. "About Microformats." http://microformats.org/about.

Oxford University Computing Services. 2010. OxGarage. http://www.oucs.ox.ac.uk/oxgarage/.

Raggett, Dave. 2003. “Clean up your Web pages with HTML TIDY.” http://www.w3.org/People/Raggett/tidy/.

Santorini, Beatrice. 1990. *Part-of-Speech Tagging Guidelines for the Penn Treebank Project*, 3rd rev. (2nd printing with updates to POS tags by Robert MacIntyre, 1995). ftp://ftp.cis.upenn.edu/pub/treebank/doc/tagguide.ps.gz.

schema.org. 2011. “Frequently asked questions.” http://schema.org/docs/faq.html.

“Stanford POS Tagger FAQ.” n.d. http://nlp.stanford.edu/software/pos-tagger-faq.shtml.

SyncRO Soft. 2010. Oxygen XML Editor. Version 12. http://oxygenxml.com/.

Toutanova, Kristina, Dan Klein, Christopher Manning, William Morgan, Anna Rafferty, Michel Galley, and John Bauer. 2011. *Stanford Log-Linear Part-of-Speech Tagger*. Version 3.0.3. http://nlp.stanford.edu/software/tagger.shtml.

Walsh, Norman. n.d. “XML Calabash.” http://xmlcalabash.com/.

Walsh, Norman, Alex Milowski, and Henry S. Thompson. 2010. “XProc: An XML Pipleine Language.” World Wide Web Consortium. http://www.w3.org/TR/xproc/.

NOTES

1. Or, if the source format is a non-XML version of HTML that can be massaged into a well-formed XML document. DuCharme (2000) and Marchal (2003) have advocated using HTML Tidy to prep non-XML-based HTML for use as a source document in an XSLT transformation. Tidy (Raggett 2003) can produce the well-formed XML that XSLT requires as input. This could be performed as a discrete step (scripted or manual) before XSLT processing or it could be done as part of an XProc (Walsh et al. 2010) pipeline using XProc’s exec step, which executes external commands. The XProc pipeline would load the HTML document, capture the result of executing Tidy on it, process the tidied document with XSLT, and output the result.

2. There are, however, obstacles to overcome. First off, there is not necessarily a one-to-one, one-to-many, or many-to-one correlation between source and target elements. Furthermore, the possibility, inherent in any process of translation, for data loss during the transformation certainly exists. After all, transformation by means of XSL Transformations (hereafter XSLT) is a selective, interpretive process; this fact becomes all the more clear when either the source or target application profile has the potential for narrative or miscellaneous prose content, such as HTML or TEI.

3. This is a matter of the TEI community’s habits and preferences, not a deficiency in the language itself. One could, in theory, build a user agent to display TEI the way web browsers display web pages built with HTML.

4. The authors make use of the metaphor of directionality in order to provide convenient labels for two kinds of transformations (TEI to HTML and HTML to TEI). We mean “forward” not to imply that such a transformation is proper or progressive but that it lies forward along the usual timeline of electronic edition publication. “Backward,” by extension, we mean not as improper or regressive but as the opposite in directionality

from a forward transformation. Furthermore, "forward," in this context, is not a synonym for "lossless."

5. RDFa (the "a" stands for "attributes") is "a collection of attributes to express structured data in any markup language" via RDF (Adida et al. 2008). It allows document authors to make RDF (Lassila and Swick 1999; Klyne and Carroll 2004; Hayes 2004) assertions about markup content directly in the host markup language (predominantly XHTML). An alternative to RDFa, for HTML5 only, is HTML Microdata (Hickson 2011). For a set of single-purpose HTML data formats, see Microformats (2011). For a primer to RDF, see Manola and Miller (2004).

6. This is not to say that all visual aspects of a web page should be effected in HTML; many aspects of presentation are the proper place of CSS.

7. It is tempting to say that it checks for the presence one or more HTML `divs` with the RDF predicate of `foaf:Person`, but this is untrue for a couple of reasons. First, as discussed later, to search for all the ways one can express an RDF predicate in RDFa, one would have to look for `@rel`, `@rev`, and `@property`. More importantly, `@typeof` always has a predicate of `rdf:type` (Adida et al. 2008 sec. 6.1.1.4.). Setting `@typeof` to `foaf:Person` does not specify a triple with a predicate of `foaf:Person`. Rather, it specifies that a blank node should be created with the predicate `rdf:type` and the object `foaf:Person`.

8. The TEI results, for the sake of brevity, are only fragments. We expect that projects making use of transforming backward would likely not generate TEI headers from the HTML source.

9. We chose PG's *Alice* because it is a readily available, permissively licensed HTML version of a narratively interesting text. We should note that this document's DOCTYPE is HTML 4.01 Transitional, so it is not encoded in XML. However, the excerpted portion, provided it were wrapped in a root element, should produce no problem for an XSLT processor.

10. The `normalize-space()` function is needed here because Widger's *Alice* contains hard returns at the end of each line.

11. Widger repeats the strategy of using `<pre>` for verse throughout *Alice*, but he uses `<pre>` for other purposes as well. These include the PG boilerplate that appears at the beginning of each of their books and for decorative indicators of scene breaks within chapters, i.e.:

```
<pre>
  *    *    *    *    *    *    *

    *    *    *    *    *    *

  *    *    *    *    *    *    *
</pre>
```

12. This template contains tokenization code adapted from a script suggested by an anonymous reviewer. We would like to thank him or her for this contribution.

13. PG does not list who produced this edition. We chose *Woman in White* because it intrigued us to match its prose style against the style of the materials on which the tagger's model was trained (see below).

14. The linguistic segment categories are discussed in section 17.1 of TEI P5 (Burnard and Bauman 2007). For more on linguistic annotation, see section 17.4. See section 15 for guidelines on encoding language corpora.

15. This was in part due to the sophistication of the available tools. POS taggers tend to perform very well on modern (especially present-day) English texts because of the readily available models that have been assembled for the language. We see no reason, however, why the conversion from HTML to TEI could be limited to matters of syntax.

16. That is, it contained lines such as: `<word wid="0" pos="RB">There</word>` It did, however, convert the HTML document's `—` to "--".

17. It is worth noting that the tagger's interpretation of this passage is not perfect. For example, it assigns JJ (adjective) instead of NN (common noun) to "high-road" (see the word with `@wid` 9). Rather than expect a POS tagger to be perfect, one should expect its inaccuracies to be statistically insignificant. In a small text such as this, one error makes a difference. However, we would expect that one would not make broad conclusions from such a small population size. (A text such as this one would only be one part of a larger corpus.) Furthermore, it is likely that this particular problem is caused by a mismatch in the eras, locations, and genres of the target text (19th century English novel) and the corpus on which this model was trained (20th century American newspaper articles). Ultimately, for the purposes of demonstrating the transformation from HTML to TEI, the accuracy of the tagger is not significant.

18. There is no punctuation class in the Penn Treebank format. Since `@type` must be an XML Name, it can only contain certain types of punctuation. Thus we normalize this to a single punctuation class for the purpose of convenience. Other researchers may choose to convert the punctuation marks to more specific classes or to use `@subtype`.

19. For example, it is unclear whether an `<li>` with `@rel='foaf:mbox'` should be treated the same as a `<p>` with `@rel='foaf:mbox'` in terms of the resulting TEI.

20. While schema.org currently supports only Microdata, they "will also be monitoring the web for RDFa and microformats adoption and if they pick up, [they] will look into supporting these syntaxes" (schema.org 2011). This strikes us an odd assertion because the big three players have to know that they will be observing the output of content creators whose decisions about whether to use Microdata or RDFa were influenced by the fact that schema.org has such broad search engine support.

ABSTRACTS

The standard workflow for preparing digital editions for display involves writing XSL to transform handcrafted TEI into either 1) HTML for the web or 2) XSL-FO for conversion into a

print friendly format such as PDF. With either method we implicitly recognize that TEI, even coupled with CSS, is not designed as a presentation technology. Many born-digital documents, however, are encoded in formats that are, such as HTML. Hypothetical future editions of such documents would most likely need to be supplemented by a document description that goes beyond the facilities of HTML to meet the needs of editors. Thus we foresee cases where born-HTML documents could be supplemented and described by TEI in much the same way as TEI currently supplements and describes manuscripts and printed books. In this paper we investigate ways that XHTML documents both with and without RDFa can be "transformed backward" into TEI. In addition to the digital edition use case, we also investigate a process for converting HTML content to TEI-based language corpora.

INDEX

Keywords: corpus linguistics, HTML, RDF, RDFa, XSL, XSLT

AUTHORS

GRANT LEYTON SIMPSON

Grant Leyton Simpson is a doctoral candidate in the Department of English at Indiana University.

DOT PORTER

Dot Porter is Associate Director for Digital Library Content and Services in the Digital Library Program at Indiana University.